Tema 2

Vectores y matrices

Guillermo Peris Ripollés

Objetivos

Cuando finalice este tema, el alumno deberá ser capaz de:

- Definir vectores y matrices con OCTAVE.
- Comprender y utilizar la notación de dos puntos para la creación de listas.
- Controlar el acceso y modificación de elementos y conjuntos de elementos de vectores y matrices.
- Realizar operaciones escalares con matrices.
- Conocer los distintos tipos de matrices predefinidas por OCTAVE.

Aplicación

Cuando finalice este tema, el alumno deberá ser capaz de resolver problemas como el siguiente, cuya resolución se indica a lo largo del propio tema.

Cinética de reacciones consecutivas

Algunas reacciones transcurren mediante la formación de un intermediario, como en las *reacciones consecutivas de primer orden* $A \rightarrow B \rightarrow C$. Las ecuaciones que rigen estas reacciones, y que proporcionan la concentración de cada especie en función del tiempo t, son las siguientes:

$$[A] = [A]_0 e^{-k_1 t}$$
$$[B] = k_1 [A]_0 \left\{ \frac{e^{-k_1 t} - e^{-k_2 t}}{k_2 - k_1} \right\}$$
$$[C] = [A]_0 \left\{ 1 + \left(\frac{k_1 e^{-k_2 t} - k_2 e^{-k_1 t}}{k_2 - k_1} \right) \right\}$$

donde $[A]$, $[B]$ y $[C]$ son las concentraciones de las respectivas especies, $[A]_0$ la concentración inicial de A y k_1 y k_2 las constantes de velocidad. Escribe un programa que pida al usuario los valores de $[A]_0$, k_1 y k_2, y calcule las concentraciones de las tres especies en función del tiempo. Comprueba el funcionamiento del programa cuando $[A]_0$ =1 mol/l, k_1 =0.1 s^{-1} y k_2 =0.2 s^{-1}.

Contenidos

2.1. Introducción

Hasta ahora únicamente hemos trabajado con números simples (más formalmente, escalares), pero ya hemos comentado que la enorme potencia de cálculo de Octave se encuentra fundamentalmente en el cálculo vectorial y matricial. En esta unidad se van a introducir estos nuevos conceptos de forma simple, aplicándolos en su parte final a la resolución de problemas relacionados con la ingeniería.

2.2. Vectores

En Octave los vectores se pueden crear introduciendo una lista de valores separados por espacios o comas y encerrados entre corchetes. Veamos un ejemplo a continuación:

```
≫t = [4 8 -2 3 5]
t =
   4    8    -2    3    5
```

En numerosas ocasiones, nos interesarán listas de valores en las que sus elementos guarden una cierta estructura, relación u orden. Por ejemplo, podríamos estar interesados en un vector con los enteros comprendidos entre 0 y 10:

```
≫t = [0 1 2 3 4 5 6 7 8 9 10]
t =
   0    1    2    3    4    5    6    7    8    9    10
```

Los valores de los elementos de t los hemos introducido uno a uno porque no son muchos pero, ¿y si hubiéramos querido introducir una lista de valores de 0 a 100?. Para facilitar esta tarea, Octave introduce la *notación de dos puntos*(:). Si escribimos dos números enteros separados por dos puntos, Octave genera todos los enteros comprendidos entre ellos. Así, podríamos crear el vector t como sigue:

```
≫t = [0:10] ;
```

Es decir, la orden `[i:j]` crea el vector `[i i+1 i+2 ... j-2 j-1 j]`. Si quisiéramos que el intervalo entre los elementos fuera distinto de 1, utilizaríamos tres números separados por ':', siendo el número central el incremento:

```
≫s = [0:2:10]
s =
     0    2    4    6    8    10
```

En este ejemplo, el vector creado contiene los valores entre 0 y 10 separados por 2 unidades. Este valor de incremento puede poseer cualquier valor entero, real e incluso negativo:

```
≫s = [10:-2:0]
s =
     10    8    6    4    2    0
```

OCTAVE también tiene definida la función `linspace(inicial,final,elementos)` que permite crear un número determinado de valores (`elementos`) distribuidos homogéneamente entre dos límites (`inicial` y `final`) . Dicho de otra forma, en lugar de especificar el intervalo entre elementos (como se hacía con la notación de dos puntos) se indica el número de elementos que queremos. Por ejemplo,

```
≫s = linspace(0,10,6)
s =
     0    2    4    6    8    10
```

crea un vector de 6 elementos equiespaciados entre 0 y 10. También en este caso se podría crear un vector de valores decrecientes cambiando el orden de los valores inicial y final:

```
≫s = linspace(10,0,6)
s =
     10    8    6    4    2    0
```

Ejercicios

▶ **1** ¿Qué vectores crean las siguientes instrucciones? Piénsalo antes de obtener el resultado con OCTAVE.

a) `[1:9]`
b) `[-1:9]`
c) `[1:2:9]`
d) `[1:0.5:9]`
e) `[1:3:9]`
f) `[9:-2:1]`
g) `[9:-3:1]`
h) `[9:-3:-1]`

▶ **2** Utiliza la función `linspace` para crear los vectores del ejercicio anterior.

2.2.1. Acceso y modificación de elementos de vectores

En ocasiones, nos interesa acceder a elementos determinados de un vector previamente definido, o bien modificar dichos elementos. Para ello, OCTAVE permite el uso de índices (expresados entre paréntesis), de forma que `t(1)` sería el primer elemento de `t`, `t(2)` el segundo, etc. Veamos un ejemplo:

```
≫t = [6 1 -1 12 4.4 -2.36 2 0.87 48 9 0] ;
≫t(1)
ans =
   6
≫t(6)
ans =
   -2.36
```

También puede utilizarse la notación de dos puntos para acceder a un conjunto de elementos, en cuyo caso el resultado es un nuevo vector.

```
≫r = t(1:3)
r =
   6    1    -1
≫s = t(10:-1:7)
s =
   9    48    0.87    2
```

En el ejemplo anterior, el vector `r` incluye los elementos de las posiciones 1-2-3 de `t`, y el vector `s` los elementos de las posiciones 10-9-8-7, en ese orden. De hecho, es posible acceder a elementos cualesquiera y en cualquier orden del vector mediante el uso de una lista de elementos. Por ejemplo, en las siguientes líneas se crea un nuevo vector `u` eligiendo los elementos de las posiciones 3, 8, 1 y 6 de `t`, en este orden:

```
≫u = t([3 8 1 6])
u =
   -1    0.87    6    -2.36
```

Mediante esta selección de elementos de un vector, también podemos modificar vectores ya creados, mediante órdenes de asignación. A continuación se muestra un ejemplo a partir del vector `t`:

```
≫t(1) = 2
t =
   2    1    -1    12    4.4    -2.36    2    0.87    48    9    0
≫t(4:6) = [5 4 3]
t =
   2    1    -1    5    4    3    2    0.87    48    9    0
≫t([11 7 10 8 9]) = [0 1 2 3 4]
t =
   2    1    -1    5    4    3    1    3    4    2    0
```

Fijémonos en que en algunas de las instrucciones anteriores se realizan asignaciones múltiples. Así, en la segunda instrucción se asignan simultáneamente 3 valores a 3 posiciones del vector `t`, mientras que en la tercera se realizan 5 asignaciones. Obviamente, si el número de elementos a ambos lados del operador de asignación es distinto, Octave no realizará ninguna asignación, indicándolo con un error.

```
≫t(4:6) = [5 4 3 2]
error: A(I) = X: X must be a scalar or a vector with the same
length as I
...........
```

Ejercicios

▶ **3** A partir del vector `v = [1 3 5 7 9 11 13 15 17 19]`, ejecuta las siguientes órdenes en OCTAVE, pensando previamente cuál va a ser el resultado.

a) `v(3)`
b) `v(3:5)`
c) `v(5:-1:3)`
d) `v(10:-2:1)`
e) `v([1 3 5 7 9])`
f) `v(v(1:5))`

▶ **4** Piensa cuál de las siguientes asignaciones es correcta, indicando en ese caso cuál sería el resultado. Utiliza el vector `v` definido en el ejercicio anterior. Compruébalo con OCTAVE.

a) `a = v(3)`
b) `b = v(3:5)`
c) `b(1:2) = v(7:9)`
d) `b(1:3) = v(7:9)`

2.3. Matrices

Denominamos matriz de dimensiones $m \times n$ a un conjunto de números ordenados en m filas y n columnas.

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}$$

Un vector no es más que un caso particular de matriz en el que una de sus dimensiones es 1. Denominaremos *vector fila* a una matriz $1 \times n$ y *vector columna* a una matriz $m \times 1$.

$$\text{Vector fila} \quad \begin{bmatrix} a_1 & a_2 & \cdots & a_n \end{bmatrix} \qquad \text{Vector columna} \quad \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_m \end{bmatrix}$$

Para introducir matrices en OCTAVE utilizaremos corchetes (al igual que con vectores), separando las filas por puntos y coma (;) y las columnas de cada fila con espacios (o comas). Las filas también pueden separarse con cambios de línea. A continuación se muestra la definición de una matriz A de dimensiones 3×4 de dos formas distintas [1]:

```
≫ A = [1 2 3 4; 5 6 7 8; 9 10 11 12]
A =
   1    2    3    4
   5    6    7    8
   9    10   11   12
```

[1] ¿Se te ocurre una forma más rápida de introducir esta matriz?

```
≫A = [1 2 3 4
      5 6 7 8
      9 10 11 12]
A =
   1    2    3    4
   5    6    7    8
   9    10   11   12
```

La función `size` aplicada sobre una matriz (o vector) devuelve un vector de dos componentes con sus dimensiones en filas y columnas:

```
≫size(A)
ans=
   3    4
```

Podemos acceder a los elementos individuales de esta matriz indicando entre paréntesis la fila y columna del elemento separadas por comas. Así,

```
≫A(2,3)
ans =
   7
```

De nuevo, la potente notación de dos puntos de Octave puede utilizarse para crear submatrices a partir de una matriz original. Considera los siguientes ejemplos, en los que se utiliza la matriz `A` definida anteriormente:

```
≫b = A(2:3,3)
b =
   7
   11
≫c = A(3,:)
c =
   9    10    11    12
≫d = A([2 3],[2 4])
d =
   6    8
   10    12
```

En la primera expresión, se crea un nuevo vector columna `b` tomando de las filas 2 y 3 los elementos de la tercera columna. En la segunda orden, utilizamos el operador ':' sin números que lo delimiten para indicar que nos interesan *todos los elementos* de la fila 3 de `A`. Por último, en la expresión final creamos una nueva matriz tomando los elementos de las posiciones 2 y 4 de las filas 2 y 3.

Al igual que se vió en el caso de vectores, también podemos modificar los elementos de una matriz con la instrucción de asignación =, siempre que las dimensiones a ambos lados del signo igual sean las mismas:

```
≫A([2 3],[2 4]) = [0 1; 2 3]
A =
 1   2   3  4
 5   0   7  1
 9   2  11  3
```

Por último, también podemos construir nuevas matrices yuxtaponiendo matrices ya existentes, siempre que éstas tengan las dimensiones adecuadas. Fíjate en los siguientes ejemplos:

```
≫X = [1 2 ; 3 4]
X =
   1    2
   3    4
≫Y = [5 6 ; 7 8]
Y =
   5    6
   7    8
≫Z = [3 4 5 6 ; 7 8 9 10]
Z =
   3    4    5    6
   7    8    9    10
≫B = [X Y]
B =
   1    2    5    6
   3    4    7    8
≫C = [X ; Y]
C =
   1    2
   3    4
   5    6
   7    8
≫D = [X Z]
D =
   1    2    3    4    5    6
   3    4    7    8    9    10
≫E = [X ; Z]
error: number of columns must match (4 != 2)
```

Ejercicios

▶ **5** Define con Octave las siguientes matrices y comprueba sus dimensiones con la función `size`.

$$A = \begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 7 & 8 & 9 & 10 & 11 & 12 \end{bmatrix} \quad B = \begin{bmatrix} 1 & 7 \\ 2 & 8 \\ 3 & 9 \\ 4 & 10 \\ 5 & 11 \\ 6 & 12 \end{bmatrix} \quad C = \begin{bmatrix} 0.6 & 1.5 & 2.3 & -0.5 \\ 8.2 & 0.5 & -0.1 & -2.0 \\ 5.7 & 8.2 & 9.0 & 1.5 \\ 0.5 & 0.5 & 2.4 & 0.5 \\ 1.2 & -2.3 & -4.5 & 0.5 \end{bmatrix}$$

▶ **6** ¿Cuál será el resultado de las siguientes expresiones en OCTAVE? Comprueba tus afirmaciones con el propio programa. Utiliza las matrices definidas en el ejercicio anterior.

a) a1 = A(2,3)
b) a2 =A(3,2)
c) b1 = B(2,3)
d) b2 =B(3,2)
e) c1 = C(2,3)
f) c2 =C(3,2)
g) a3 = A(:,2)
h) c3 = C(4:5,1:3)
i) c4 = C(1:2:5,:)
j) c5 =C([5 2 1], 3:-1:2)
k) D = [4:9; 1:6]
l) E = [D A]
m) A(:,2:3) = B(1:2,:)
n) A(1,4:6) = C(4,1:3)
ñ) B(4:6,2) = B(4:6,1)
o) C(1,:) = A(1,1:4)

2.4. Matrices predefinidas

La definición de matrices mediante la introducción elemento a elemento es costosa cuando se trata de matrices de gran tamaño. Para facilitar la labor de crear matrices, OCTAVE dispone de una serie de funciones predefinidas:

- **Matriz de ceros** : La función `zeros` genera una matriz que contiene 0's en todos sus elementos. Si la función sólo recibe un argumento (`zeros(m)`) se genera una matriz cuadrada de m filas, mientras que si se le pasan dos (`zeros(m,n)`) se genera una matriz rectangular de dimensiones $m \times n$. En este último caso, también puede utilizarse la salida de la función `size` para crear una matriz de ceros cuyas dimensiones vendrán dadas por otra matriz. Todos estos conceptos se entenderán mejor con los siguientes ejemplos:

```
≫A = zeros(3)
A =
   0   0   0
   0   0   0
   0   0   0
≫B = zeros(3,2)
B =
   0   0
   0   0
   0   0
≫C = [ 1 2 3 ; 4 2 5]
C =
   1   2   3
   4   2   5
≫D = zeros(size(C))
D =
   0   0   0
   0   0   0
```

- **Matriz de unos** : Exactamente igual que la función `zeros`, pero ahora se genera una matriz con elementos unidad. El nombre de la función es `ones`.

```
≫A = ones(3,2)
A =
   1   1
   1   1
   1   1
```

- **Matriz identidad** : La función `eye` crea una matriz en la que todos los elementos son ceros a excepción de los elementos diagonales (aquellos en que los índices de filas y columnas son iguales) que son unos. A esta matriz se le conoce como matriz identidad y se denota matemáticamente como $\mathbb{I}$. Los argumentos de esta función son similares a los de las funciones anteriormente definidas. Aunque, formalmente, las matrices identidad son cuadradas, esta función de OCTAVE permite la creación de matrices identidad rectangulares.

```
≫A = eye(3)
A =
   1   0   0
   0   1   0
   0   0   1
≫B = eye(3,2)
B =
   1   0
   0   1
   0   0
≫C = [ 1 2 3 ; 4 2 5]
C =
   1   2   3
   4   2   5
≫D = eye(size(C))
D =
   1   0   0
   0   1   0
```

- **Matriz diagonal**: Se denomina así a aquella matriz en la que todos los elementos extradiagonales son ceros, es decir, únicamente presentan valores distintos de cero los elementos de la diagonal. OCTAVE nos permite crear una matriz diagonal a partir de un vector, que situa en la diagonal de la matriz. Para ello utilizaremos la función `diag` como sigue:

```
≫X = [4 3 2 1];
≫Y = diag(X)
Y =
   4   0   0   0
   0   3   0   0
   0   0   2   0
   0   0   0   1
```

Por el contrario, si el argumento de la función es una matriz, `diag` extrae el vector que contiene los elementos diagonales:

```
≫A = [ 1 2 3; 4 5 6; 7 8 9]
A =
   1   2   3
   4   5   6
   7   8   9
≫diag(A)
ans =
   1
   5
   9
```

- **Matriz aleatoria**: La función `rand(n)` crea una matriz de dimensión $n \times n$ con elementos aleatorios distribuidos uniformemente entre 0 y 1, mientras que con dos argumentos (`rand(m,n)`) crea una matriz de dimensiones $m \times n$.

```
≫A = rand(4,5)
A =
   0.722684   0.678103   0.228702   0.648577   0.124333
   0.336018   0.976399   0.458428   0.525437   0.069060
   0.651192   0.483592   0.744526   0.952700   0.229671
   0.464437   0.166825   0.163426   0.568459   0.038776
```

Ejercicios

▶ **7** Haciendo uso de las funciones vistas hasta ahora, define las siguientes matrices:

a) $\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$

b) $\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix}$

c) $\begin{bmatrix} 7 & 0 & 0 & 0 \\ 0 & 9 & 0 & 0 \\ 0 & 0 & 11 & 0 \\ 0 & 0 & 0 & 13 \end{bmatrix}$

d) $\begin{bmatrix} 2 & 1 & 1 & 1 \\ 1 & 2 & 1 & 1 \\ 1 & 1 & 2 & 1 \\ 1 & 1 & 1 & 2 \end{bmatrix}$

▶ **8** Dada una matriz `A` cualquiera, ¿a qué es igual la operación `diag(diag(A))`? Para comprobar tu respuesta, aplícalo a una matriz A aleatoria de dimensiones 3×3 creada con la función `rand`.

2.5. Operaciones con matrices

2.5.1. Transposición

La matriz transpuesta de una matriz A de dimensión $m \times n$ es una matriz de $n \times m$ en la que el elemento (i, j) ha pasado a la posición (j, i), o dicho coloquialmente, se han intercambiado las filas y columnas de A. Por ejemplo, la matriz transpuesta de

$$\begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \end{bmatrix} \quad \text{es} \quad \begin{bmatrix} 1 & 5 \\ 2 & 6 \\ 3 & 7 \\ 4 & 8 \end{bmatrix}$$

En Octave, la transposición de una matriz se efectua añadiendo el apóstrofe (') al nombre de la matriz [2].

```
≫A= [1 2 3 4; 5 6 7 8];
≫B = A'
B =
   1   5
   2   6
   3   7
   4   8
```

Observamos que con la transposición se convierten vectores fila en vectores columna y viceversa. Esta característica puede ser muy útil para mostrar vectores en forma de tabla. Imaginemos, por ejemplo, que de un experimento de cinética de reacción con medidas de conductividad obtenemos un vector `t` de tiempos, `conc` de concentraciones y `rho` de conductividades. Podemos visualizar una tabla de valores del siguiente modo:

```
≫t = [1:5];
≫conc = [0.5 0.3 0.05 0.01 0.00005];
≫rho = [27.6 15.4 2.89 0.56 0.026];
≫[t' conc' rho']
ans =
   1.0000   0.50000   27.6000
   2.0000   0.30000   15.4000
   3.0000   0.05000   2.8900
   4.0000   0.01000   0.5600
   5.0000   0.00005   0.0260
```

2.5.2. Aritmética escalar

En Octave se denominan *operaciones escalares* a aquellas que actuan individualmente sobre cada uno de los elementos de una matriz o vector. Forman parte de esta categoría las operaciones mixtas binarias en las que intervienen una matriz y un escalar. Por ejemplo, si queremos multiplicar por un mismo valor todos los elementos de una matriz, o bien sumarles dicho valor, realizaríamos las siguientes operaciones:

```
≫A = [1 2 3 4; 5 6 7 8];
≫B = 2*A
B =
  2   4   6   8
 10  12  14  16
```

```
≫C = B - 2
C =
 0   2   4   6
 8  10  12  14
```

[2] En el teclado el apóstrofe suele estar en la tecla a la derecha del 0.

La matriz `B` se construye multiplicando por 2 cada uno de los elementos de `A`, mientras que C se define restando 2 a los elementos de `B`.

Pero, ¿qué ocurre si queremos multiplicar dos matrices de dimensiones semejantes elemento a elemento?. En ese caso no podemos utilizar el operador '*', ya que éste realizaría un *producto matricial* (explicado en una sección posterior), cuyo resultado sería distinto al esperado. En general, para realizar operaciones escalares con matrices utilizaremos los operadores antecedidos de un punto. Para entenderlo mejor, vamos a seguir una serie de ejemplos utilizando vectores:

```
≫A = [2 5 6] ;
≫B = [2 3 5] ;
≫C= A.*B
C =
   4    15    30
≫D = A./B
D =
   1.0000    1.6667    1.2000
```

Las sumas y restas de matrices (siempre que sus dimensiones sean las mismas) se consideran en el algebra matricial como operaciones elemento a elemento (al contrario que las multiplicaciones y divisiones) por lo que, aunque podemos utilizar los operadores `.+` y `.-`, el punto no es necesario.

```
≫A + B
ans =
   4    8    11
≫A - B
ans =
   0    2    1
```

También pueden realizarse exponenciaciones elemento a elemento, como se muestra en el siguiente ejemplo:

```
≫C= A.^2
C =
   4    25    36
≫D= 3.^A
D =
   9    243    729
≫E = A.^B
E =
   4    125    7776
```

Así pues, en general, sólo será necesario anteceder el operador con un punto en el caso de multiplicaciones, divisiones y exponenciaciones cuando se pretenda realizar una operación elemento a elemento.

Estas operaciones, además de ser válidas para vectores, también lo son para matrices, como se observa en los siguientes ejemplos:

```
≫d = [1:5; -1:-1:-5]
d =
   1    2    3    4    5
  -1   -2   -3   -4   -5
≫p = d.*5
p =
   5    10    15    20    25
  -5   -10   -15   -20   -25
≫q = d.^3
q =
   1    8    27    64    125
  -1   -8   -27   -64   -125
```

Ejercicios

▶ **9** Calcula los valores del vector `C` tras ejecutar las órdenes que se indican, utilizando los vectores `A = [2 -1 5 0]` y `B = [3 2 -1 4]`. Comprueba tus respuestas con Octave.

a) `C = B + A -3;`

b) `C = A./B;`

c) `C = A.^B;`

d) `C = 2*A + A.^B;`

e) `C = 2.^B + A;`

f) `C = 2*B/3.*A;`

▶ **10** Crea un vector `x` con los siguientes elementos:

a) 1, 1/2, 1/3, 1/4, 1/5, 1/6

b) 0, 1/2, 2/3, 3/4, 4/5

2.5.3. Aplicación de funciones sobre matrices

Las funciones matemáticas de Octave actuan sobre las matrices del mismo modo que las operaciones escalares, es decir, se aplican elemento a elemento. Veamos un ejemplo:

```
≫x = pi*[0:0.1:0.9];
≫y = cos(x)
y =
   Columns 1 through 5:
   1.0000e+00 9.5106e-01 8.0902e-01 5.8779e-01 3.0902e-01
   Columns 6 through 10:
   6.1230e-17 -3.0902e-01 -5.8779e-01 -8.0902e-01 -9.5106e-01
```

```
≫z = abs(y)
z =
   Columns 1 through 5:
   1.0000e+00 9.5106e-01 8.0902e-01 5.8779e-01 3.0902e-01
   Columns 6 through 10:
   6.1230e-17 3.0902e-01 5.8779e-01 8.0902e-01 9.5106e-01
```

Así, el elemento i del vector y contiene el coseno del elemento i del vector x, y el de **z** su valor absoluto.

Ejercicios

11 Dado el vector `t = [1:0.2:2]`, calcula las siguientes expresiones matemáticas con Octave:

a) $\ln(2 + t + t^2)$

b) $e^t(1 + \cos(3t))$

c) $\cos^2(t) + \sin^2(t)$

d) $arctg(t)$

e) $cotg(t)$

f) $\sec^2(t) + cotg(t) - 1$

Además, Octave dispone de una serie de funciones que actuan de forma específica sobre vectores y matrices. Vamos a considerar aquí algunas de ellas, de interés en cálculos estadísticos. En particular, vamos a destacar el uso de funciones para el cálculo de máximos y mínimos, sumas y productos de elementos, medias y ordenaciones. La actuación de estas funciones depende de que sus argumentos sean vectores y funciones, tal y como se explica en la Tabla 2.1.

Función	Aplicación sobre	
	Vectores	Matrices
`max(x)`	Valor mayor en `x`	Vector fila con elemento máximo de cada columna
`min(x)`	Valor menor en `x`	Vector fila con elemento mínimo de cada columna
`length(x)`	Número de elementos en `x`	Dimensión máxima de la matriz
`sum(x)`	Suma de los elementos de `x`	Vector fila con la suma de los elementos de cada columna
`prod(x)`	Producto de los elementos de `x`	Vector fila con el producto de los elementos de cada columna
`mean(x)`	Media de los elementos de `x`	Vector fila con la media de los elementos de cada columna
`std(x)`	Desviación estándar de los elementos de `x`	Vector fila con la desviación estándar de los elementos de cada columna
`median(x)`	Mediana de los elementos de `x`	Vector fila con la mediana de los elementos de cada columna
`sort(x)`	Ordena el vector `x` de forma ascendente	Devuelve la matriz con los elementos de cada columna ordenados de forma ascendente

Tabla 2.1: Algunas funciones aplicables sobre vectores y matrices.

Utilicemos vectores y matrices concretos para entender mejor la acción de estas funciones.

```
≫v = [3 4 1 9 5 6]
v =
 3  4  1  9  5  6
≫M = [0.6 1.5 2.3 -0.5 ; 8.2 0.5 -0.1 -2.0; 5.7 8.2 9.0 1.5 ; ...
0.5 0.5 2.4 0.5 ; 1.2 -2.3 -4.5 0.5]
M =
 0.6   1.5   2.3  -0.5
 8.2   0.5  -0.1    -2
 5.7   8.2     9   1.5
 0.5   0.5   2.4   0.5
 1.2  -2.3  -4.5   0.5
≫max(v)
ans = 9
≫max(M)
ans =
 8.2  8.2  9  1.5
```

La función `max` devuelve, al ser aplicada a un vector, el valor máximo del mismo, en este caso 9. Sin embargo, al tener como argumento una matriz devuelve un vector fila en el que cada elemento es el máximo de la columna correspondiente de la matriz original. La función `sum` presenta un comportamiento similar con vectores y matrices, sumando las componentes de un vector y las componentes de cada columna en una matriz, como se muestra en el siguiente ejemplo:

```
≫sum(v)
ans = 28
≫sum(M)
ans =
 16.2  8.4  9.1  0
```

La función `length` calcula la longitud del vector, o sea, el número de elementos que contiene. En el caso de una matriz, devuelve el número máximo de filas o columnas, es decir, la dimensión de la matriz de mayor valor.

```
≫length(v)
ans =
   6
```

```
≫length(M)
ans =
   5
```

La función `mean` calcula la media de los elementos del vector, es decir, la suma de sus elementos dividida por el número de ellos. Esto se aplica sobre las columnas al tener como argumento una matriz.

```
≫mean(v)
ans =
   4.6667
≫mean(M)
ans =
 3.24   1.68   1.82   0
```

Por último, la función `sort` ordena los elementos de un vector, mientras que en el caso de matrices ordena de forma independiente cada una de sus columnas.

```
≫sort(v)
ans =
 1   3   4   5   6   9
≫sort(M)
ans =
 0.5   -2.3  -4.5    -2
 0.6    0.5  -0.1  -0.5
 1.2    0.5   2.3   0.5
 5.7    1.5   2.4   0.5
 8.2    8.2     9   1.5
```

Ejercicios

▶ **12** Determina las matrices devueltas por las siguientes llamadas a función, verificando posteriormente tus respuestas con Octave. Deberás definir previamente las siguientes variables:

$$w = \begin{bmatrix} 0 & 3 & -2 & 7 \end{bmatrix} \quad x = \begin{bmatrix} 3 & -1 & 5 & 7 \end{bmatrix} \quad y = \begin{bmatrix} 1 & 3 & 7 \\ 2 & 8 & 4 \\ 6 & -1 & -2 \end{bmatrix}$$

a) max(w)
b) min(y)
c) mean(y)
d) median(w)
e) sort(2*w+x)
f) sort(y)
g) length(w)
h) length(y)

▶ **13** En Octave, ¿es equivalente `mean(x)` y `sum(x)/length(x)`? Compruébalo con los siguientes vector y matriz y trata de dar una explicación a tus observaciones.

$$w = \begin{bmatrix} 0 & 3 & -2 & 7 \end{bmatrix} \quad y = \begin{bmatrix} 1 & 3 & 7 \\ 2 & 8 & 4 \end{bmatrix}$$

▶ **14** ¿Cómo calcularías el valor máximo de entre todos los elementos de una matriz?

2.6. Aritmética matriz-matriz

Vamos a tratar ahora las operaciones internas entre matrices, que son operaciones matemáticas en que ambos operadores son matrices. Aunque la suma y resta de matrices son operaciones elemento a elemento, las repasamos aquí por encontrarse dentro de las operaciones entre matrices.

2.6.1. Suma y resta

La suma de dos matrices únicamente es posible si éstas tienen las mismas dimensiones. Es decir, la función `size` aplicada a ambas ha de proporcionar el mismo resultado. La matriz suma es una matriz de las mismas dimensiones en la que cada uno de sus elementos es igual a la suma de los elementos que ocupan la misma posición en las matrices sumando. Por ejemplo,

$$\begin{bmatrix} 2 & 1 \\ 4 & 6 \end{bmatrix} + \begin{bmatrix} 4 & 2 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 6 & 3 \\ 4 & 7 \end{bmatrix}$$

En Octave, esta suma se llevaría a cabo como sigue:

```
≫A = [2 1 ; 4 6] ;
≫B = [4 2 ; 0 1] ;
≫C = A + B
C =
   6   3
   4   7
```

La resta se define y trata en Octave de forma equivalente.

2.6.2. Producto escalar

Podemos definir el producto escalar entre un vector fila X y un vector columna Y de la misma longitud como sigue:

$$X \cdot Y = [x_1 x_2 x_3 \ldots x_n] \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_n \end{bmatrix} = x_1 \cdot y_1 + x_2 \cdot y_2 + \ldots + x_n \cdot y_n = \sum_{i=1}^{n} x_i \cdot y_i \, .$$

Por ejemplo, para calcular el producto escalar entre los vectores `A =[2 5 -2]` y `B = [4 -3 2]` ejecutaríamos el siguiente conjunto de órdenes:

```
≫A = [2 5 -2];
≫B = [4 -3 2];
≫A*B'
ans =
   -11
```

El resultado puede obtenerse fácilmente a mano ($2*4+5*(-3)+(-2)*2 = -11$). Fijémonos en que `A` es un vector fila, y que hemos tenido que transponer el vector

antes del producto para convertirlo en un vector columna. Si no lo hubiéramos hecho, habríamos obtenido un mensaje de error[3]:

```
≫ A*B
??? Error using ==>*
Inner matrix dimensions must agree.
```

El producto escalar de dos vectores es muy importante en algebra vectorial y en física.

Ejercicios

▶ **15** A partir de los siguientes vectores, indica cuáles de las siguientes operaciones son correctas, calculando su valor.

$$A = \begin{bmatrix} 5 & 3 & -1 \end{bmatrix} \qquad B = ones(1,3) \qquad C = \begin{bmatrix} 9 & 4 & 0 \end{bmatrix}$$

$$D = eye(1,4) \qquad E = \begin{bmatrix} -1 & 3 & -5 & 7 \end{bmatrix}$$

a) `A*B`
b) `A*B'`
c) `B*A'`
d) `A*C'`
e) `B*C'`
f) `B*D'`
g) `E*D'`

2.6.3. Multiplicación

La multiplicación de matrices no es una operación elemento a elemento, es decir, no se efectúa multiplicando los elementos correspondientes de las matrices operando. Si realizamos la multiplicación $C = A \times B$, el elemento c_{ij} no se calcula como $a_{ij} \times b_{ij}$, sino como el producto escalar de los vectores correspondientes a la fila i de la matriz A y la columna j de la matriz B. En la figura 2.1 se muestra gráficamente esta operación [4]. Aquí se observa que si queremos multiplicar dos matrices A $(m \times n)$ y B $(p \times q)$, el número de columnas de A debe coincidir con el número de filas de B, es decir, $n = p$.

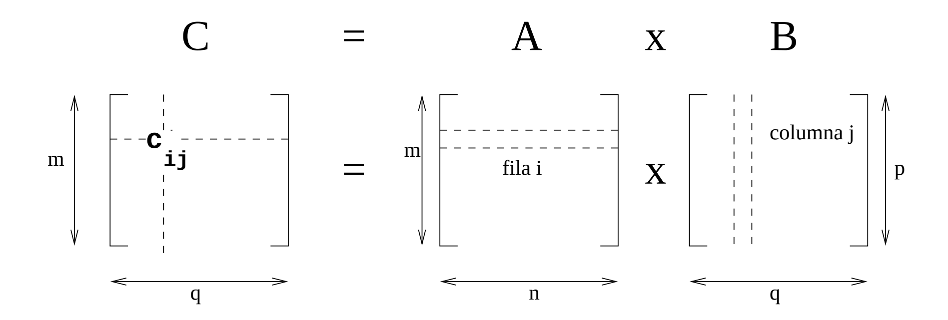


Figura 2.1: Multiplicación de matrices.

Una forma sencilla de decidir si podemos multiplicar dos matrices es escribir las tallas de las dos matrices juntas. De esta forma, si las dos dimensiones interiores coinciden, las matrices se pueden multiplicar y la dimensión de la matriz resultante coincide

[3] Podríamos haber obtenido el resultado correcto utilizando la siguiente expresión: `sum(A.*B)`.
[4] ¡Cuidado!. De esta definición se infiere que el producto de matrices no es conmutativo, es decir, $A \times B \neq B \times A$.

con las dimensiones externas. Por ejemplo, supongamos que queremos multiplicar una matriz de 2×3 por otra de 3×3 (en ese orden).

$$\overbrace{2 \times \underbrace{3\ ,\ 3}} \times 3$$

Al ser los dos números internos iguales, la multiplicación es válida, y las dimensiones de la matriz resultante son los números externos, es decir, 2×3. Consideremos las matrices siguientes:

$$A = \begin{bmatrix} 2 & 5 & 1 \\ 0 & 3 & -1 \end{bmatrix} \qquad B = \begin{bmatrix} 1 & 0 & 2 \\ -1 & 4 & -2 \\ 5 & 2 & 1 \end{bmatrix}$$

El producto de estas matrices se realizaría como sigue:

$$\begin{aligned} C &= A \times B = \begin{bmatrix} 2 & 5 & 1 \\ 0 & 3 & -1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 2 \\ -1 & 4 & -2 \\ 5 & 2 & 1 \end{bmatrix} \\ &= \begin{bmatrix} 2 \cdot 1 + 5 \cdot (-1) + 1 \cdot 5 & 2 \cdot 0 + 5 \cdot 4 + 1 \cdot 2 & 2 \cdot 2 + 5 \cdot (-2) + 1 \cdot 1 \\ 0 \cdot 1 + 3 \cdot (-1) + (-1) \cdot 5 & 0 \cdot 0 + 3 \cdot 4 + (-1) \cdot 2 & 0 \cdot 2 + 3 \cdot (-2) + (-1) \cdot 1 \end{bmatrix} \\ &= \begin{bmatrix} 2 & 22 & -5 \\ -8 & 10 & -7 \end{bmatrix} \end{aligned}$$

En Octave, esta operación se realizaría de forma sencilla con el operador de multiplicación:

```
>>A = [2 5 1; 0 3 -1];
>>B = [1 0 2; -1 4 -2; 5 2 1] ;
>>C = A*B
C =
     2  22  -5
    -8  10  -7
>>D = B*A
??? Error using ==>*
Inner matrix dimensions must agree.
```

Fijémonos en que Octave no permite el producto $B \times A$ por no ser adecuadas las dimensiones de las matrices.

Recordemos que si `A` es una matriz, `A.^2` es la operación que eleva al cuadrado cada elemento de la matriz. Si queremos calcular propiamente el cuadrado de la matriz, es decir, `A*A`, podemos ejecutar la orden `A^2`. De este modo, `A^4` es equivalente a `A*A*A*A`. Si tenemos en cuenta que en el producto de matrices el número de columnas del primer operando ha de ser igual al número de filas del segundo, deducimos que únicamente podemos obtener potencias de matrices cuadradas.

```
>>B^3
ans =
     27  24   18
    -87  24  -66
     33  54    3
```

Fijémonos en que el resultado es distinto al que se obtiene con el operador punto correspondiente:

```
≫B.^3
ans =
     1    0    8
    -1   64   -8
   125    8    1
```

Ejercicios

✍ 💾 ▶ **16** Calcula los siguientes productos de matrices, pensando previamente cuál va a ser el resultado.

$$A = \begin{bmatrix} 2 & 1 \\ 0 & -1 \\ 3 & 0 \end{bmatrix} \qquad B = \begin{bmatrix} 1 & 3 \\ -1 & 5 \end{bmatrix} \qquad C = \begin{bmatrix} 3 & 2 \\ -1 & -2 \\ 0 & 2 \end{bmatrix} \qquad D = \begin{bmatrix} 1 & 2 \end{bmatrix}$$

a) `A*B`
b) `A*C`
c) `A*C'`
d) `D*B`
e) `D*B^2`
f) `(C*B)*D'`
g) `B*C'`
h) `A*D`

2.6.4. Determinantes

El cálculo del determinante de una matriz cuadrada se realiza con la función `det`. Los determinantes son números calculados a partir de los elementos de una matriz, con amplias aplicaciones en cálculos matemáticos, físicos y de ingeniería. Matemáticamente, el determinante de una matriz A se denota como $|A|$.

```
≫A = [1 0 2; -1 4 -2; 5 2 1] ;
≫det(A)
ans =
  -36
```

2.6.5. Matriz inversa

La inversa de una matriz cuadrada A es aquella matriz que, multiplicada por la matriz original, proporciona la matriz identidad, es decir, $A \times A^{-1} = A^{-1} \times A = \mathbb{I}$. Se puede demostrar que únicamente tienen inversa aquellas matrices cuadradas cuyo determinante es distinto de cero.

Por ejemplo, consideremos la siguiente definición de dos matrices A y B:

```
≫A = [2 1; 4 3]
A =
   2    1
   4    3
≫B = [1.5 -0.5; -2 1]
B =
```

```
  1.5000   -0.5000
  -2.0000   1.0000
≫A*B
ans =
  1   0
  0   1
```

```
≫B*A
ans =
  1   0
  0   1
```

Observamos que $A \cdot B = B \cdot A$ y que este producto es igual a la matriz identidad de dimensión dos, por lo que A es la matriz inversa de B, y viceversa. A continuación comprobamos esta afirmación con OCTAVE , calculando la inversa de A, comprobando previamente que el valor de su determinante no es cero.

```
≫det(A)
ans =
  2
≫inv(A)
ans =
  1.5   -0.5
  -2   1
```

Si definimos una matriz cuyo determinante es cero, OCTAVE genera un aviso al tratar de calcular su inversa, aunque devuelve una matriz que resulta incorrecta:

```
≫C = [2 1; 4 2]
C =
  2   1
  4   2
≫det(C)
ans =
  0
≫inv(C)
warning: inverse: matrix singular to machine precision, rcond = 0
ans =
  4.00000   2.00000
  0.50000   0.00000
```

Ejercicios

▶ **17** Calcula los determinantes y, en su caso, las matrices inversas, de las siguientes matrices:

a) $\begin{bmatrix} 2 & 1 \\ 4 & 2 \end{bmatrix}$ b) $\begin{bmatrix} 4 & -3 & 2 \\ 8 & -12 & -5 \\ 5 & 9 & -2 \end{bmatrix}$ c) $\begin{bmatrix} 0.1 & -5.0 & 3.0 & 8.7 \\ 2.0 & -1.6 & 4.5 & 8.9 \\ 2.7 & 9.2 & 5.6 & 6.7 \\ 0.2 & -4.5 & -8.0 & 1.0 \end{bmatrix}$

✍ ▶ **18** En la siguiente tabla se muestra el coste por hora de cuatro procesos de manufactura distintos. También aparecen el número de horas de cada proceso necesario para fabricar tres productos diferentes.

		Horas necesarias por unidad		
Proceso	**Coste/hora (€)**	**Producto 1**	**Producto 2**	**Producto 3**
Torneado	10	6	5	4
Afilado	12	2	3	1
Molienda	14	3	2	5
Soldadura	9	4	0	3

Escribe un programa OCTAVE que,

a) determine el coste unitario de cada uno de los productos,

b) calcule el coste total de producción de 10, 5 y 7 unidades de los productos 1, 2 y 3, respectivamente.

Deberás utilizar únicamente productos de vectores y matrices.

2.7. Aplicación

A priori, la idea para resolver la aplicación planteada al principio del tema consiste en calcular con las fórmulas que se indican los valores de $[A]$, $[B]$ y $[C]$ para distintos valores del tiempo. Ahora bien, ¿hasta que valor del tiempo debemos calcular las concentraciones?.

Teniendo en cuenta que, según la reacción, el reactivo A va a desaparecer, podemos pensar que basta con esperar a que, por ejemplo, sólo quede el 1 % de A. Es muy fácil calcular a qué tiempo corresponde esta concentración. A partir de la ecuación para la concentración de $[A]$, tenemos:

$$[A] = 0.01 * [A]_0 \longrightarrow 0.01 * [A]_0 = [A]_0 * e^{-k1*t_{max}} \longrightarrow t_{max} = -\frac{ln(0.01)}{k_1}$$

Con este valor de t_{max}, resulta sencillo desarrollar el siguiente programa OCTAVE para resolver el problema planteado.

```
%*******************************************************************
% Programa : cineticaABC.m
% Descripcion: Este programa calcula y representa la variacion de
%            las concentraciones con el tiempo en una cinetica
%            del tipo A->B->C, con constantes k1 y k2.
%*******************************************************************

   % Pedimos al usuario los valores de A0, k1 y k2.
A0 = input('Introduzca el valor inicial de la especie A: ');
k1 = input('Introduzca el valor de la constante k1: ') ;
k2 = input('Introduzca el valor de la constante k2: ') ;

   % Calculamos el tiempo maximo y el vector de tiempos
tmax = round(-log(0.01)/k1) ;
t = linspace(0,tmax,100) ;

   % Calculamos los valores de las concentraciones de A,B y C.
A = A0*exp(-k1*t) ;
B = k1*A0*( exp(-k1*t) - exp(-k2*t))/(k2-k1) ;
C = A0*(1 + ( k1*exp(-k2*t) - k2*exp(-k1*t))/(k2-k1) );

% Impresion de resultados
disp('t A B C')
disp([t' A' B' C'])
```

Ejercicios

▶ **19** Escribe este programa en un fichero de nombre `cineticaABC.m` y ejecútalo con los valores que se indican en la aplicación.

2.8. Ejercicios prácticos

Es conveniente que pienses y realices los ejercicios que han aparecido a lo largo de la unidad marcados con el símbolo ✍ antes de acudir a la sesión de prácticas correspondiente. Deberás iniciar la sesión realizando los ejercicios marcados con el símbolo 💾. A continuación, deberás hacer el mayor número de los ejercicios siguientes.

Ejercicios

▶ **20** Crea un vector `x` con los elementos $x_n = \frac{(-1)^n}{2n-1}$ para $n = 1, 2, 3, \ldots, 20$ y comprueba que $x_n \to 0$ a medida que n aumenta.

▶ **21** Dada la matriz `A = [2 4 1; 6 7 2; 3 5 9]`, escribe las órdenes necesarias para

a) asignar la primera fila de `A` a un vector de nombre `x1`.

b) asignar las 2 últimas filas de `A` a una matriz de nombre `y`.

c) calcular la suma de cada una de las columnas de `A`.

d) calcular la suma de cada una de las filas de `A`.

▶ **22** Dados los vectores `x = [1 4 8]` e `y = [2 1 5]`, y la matriz `A = [3 1 6; 5 2 7]`, determina cuáles de las siguientes órdenes se ejecutarán correctamente en Octave y da su resultado. En caso de que la expresión dé un error, indica por qué.

a) `x + y`

b) `x + A`

c) `x' + y'`

d) `[x ; y']`

e) `[x ; y]`

f) `A - [x ; y]`

▶ **23** A partir de la matriz `A = [2 7 9 7; 3 1 5 6; 8 1 2 5]`, explica los resultados de las siguientes expresiones:

a) `A'`

b) `A(:,[1 4])`

c) `A([2 3],[3 1])`

d) `A(1:3,:)`

e) `[A; A(1:2,:)]`

f) `sum(A)`

g) `sum(A')`

h) `prod(A)`

i) `prod(A')`

j) `sum(sum(A))`

k) `mean(A)`

l) `mean(A')`

m) `mean(mean(A))`

▶ **24** Define las variables $m = 10$, $n = 6$, $r = 3$ y $s = 6$, y construye las siguientes matrices:

1. Matriz identidad de orden n.
2. Matriz con elementos todos igual a -1 de orden $s \times r$.
3. Matriz cuadrada con elementos de la diagonal igual a 0 y el resto 1 de orden s.

▶ **25** Define los siguientes vectores:

$$v1 = \begin{bmatrix} 1 & 2 & 3 & \cdots & 10 \end{bmatrix} \qquad v2 = \begin{bmatrix} 20 & 18 & 16 & \cdots & 2 \end{bmatrix}$$

Analiza el comportamiento de la siguientes operaciones diciendo de qué tipo de operación se trata: vectorial, escalar, matricial, elemento a elemento, ... También indica las que producen error indicando por qué:

a) $v1 + v2$
b) $v1 * v2$
c) $v1' * v2$
d) $v1. * v2$
e) $v1. \wedge v2$
f) $v1/v2$
g) $v1./v2$
h) $v1 \wedge 2$
i) $v1. \wedge 2$
j) $2 * (v1 + v2)$
k) $sin(\pi * v1). * v2$
l) $(v1'). * (v2')$
m) $(v1' * v2) \wedge (-1)$
n) $((v1'). * v2). \wedge (-1)$

▶ **26** El factor de compresibilidad z mide la desviación del comportamiento de un gas respecto a la ecuación de estado de los gases ideales. Si el gas fuese ideal, z sería igual a la unidad bajo todas las condiciones, pero lo más habitual es que este factor se desvíe considerablemente de este valor. Además las desviaciones de la conducta ideal pueden hacer que z sea mayor o menor que la unidad.

Una posible forma de calcular el factor de compresibilidad es utilizar la ecuación de estado del virial, según la cual:

$$z = 1 + \frac{a_1}{V} + \frac{a_2}{V^2} + \frac{a_3}{V^3} + \ldots$$

Las cantidades a_1, a_2, etc, se denominan el segundo, tercero,etc., coeficientes del virial, y dependen sólo de la temperatura y de las propiedades de las moléculas del gas.

Escribe un Programa que pida al usuario los coeficientes del virial y el volumen de un gas y calcule el factor de compresibilidad z.

Nota: Puedes escribir el programa sin utilizar vectores o con ellos. Puedes pedir al usuario los datos que consideres oportunos. Ten en cuenta que no asumimos a priori el número de coeficientes del virial.

▶ **27** La *media armónica* es otra forma de calcular la media de un conjunto de N números. Esta media se calcula con la siguiente expresión:

$$\text{media armónica} = \frac{N}{\frac{1}{x_1} + \frac{1}{x_2} + \cdots + \frac{1}{x_N}}$$

Escribe un programa Octave que lea un número arbitrario de valores positivos y calcule su media armónica. En caso de que aparezca algún número negativo o nulo, el programa terminará con un mensaje de error. Puedes elegir el método que quieras para pedir los datos al usuario.